Tradeoff between Performance Improvement and Energy Saving in Mobile Cloud Offloading Systems

Huaming Wu, Qiushi Wang and Katinka Wolter
Department of Mathematics and Computer Science
Free University of Berlin
Takustr.9, 14195 Berlin, Germany
Email: {huaming.wu, qiushi.wang and katinka.wolter}@fu-berlin.de

***Abstract*—Cloud computation offloading is a promising method that sending heavy computation to resourceful servers on cloud and then receiving the results from them. In this paper, we study the offloading techniques and further explore the tradeoff between shortening execution time and extending battery life of mobile devices. A novel adaptive offloading scheme is proposed and analyzed based on the tradeoff analysis. And it can be realized thanks to the elasticity of cloud computing that the resources can be bought on demand. We have tried to find a server on cloud with a critical value of speedup F for a specified mobile device. When satisfying the requirement such as performance improvement by the system, it is worth sacrificing large F when taking economic factor into consideration.**

***Index Terms*—offloading; energy; time; tradeoff; cloud computing**

I. Introduction

Recently, the issues of energy saving and performance improvement on mobile devices are becoming more and more concerned [1, 3, 5]. It is challenging to run very complex applications on the mobile devices because of the strict constraints on their resources such as memory capacity, network bandwidth, CPU speed and battery power [1]. This is not just a temporary limitation of current mobile hardware technology, but is intrinsic to mobility.

Cloud computing is becoming increasingly popular these days due to its features like elasticity, scalability and inexpensive. Along with the maturity of cloud computing [2], offloading the program from mobile devices to cloud is one of the attractive ways to overcome the above problems. Offloading brings many potential benefits, such as energy saving, performance improvement, reliability improvement, ease for the software developers, better exploitation of contextual information and so on. The main advantage of cloud computing is its elastic execution that resources can be obtained on demand, it is an enabler for computation offloading. Therefore, we can choose the server on cloud that required by users to offload the program on mobile devices. Cloud offloading has the potential to save execution time and energy consumption but the savings from offloading the computation need to exceed the time and energy cost due to the additional communication between mobile devices and cloud [3]. The issue of how to make offloading decision has been extensively explored but is still under widely investigation since it's combined by cloud computing technique.

In our analysis, we discuss the tradeoff between reducing execution time and saving energy consumption in cloud offloading systems. When considering the economic factor, the tradeoff point is strongly dependent on a speedup factor. For a specified mobile device, it can be found that a server with a critical value of speedup on cloud. Therefore, when meeting performance improvement required by the system, it is worth sacrificing energy consumption in order to save money and resources.

Accordingly, the main contributions of our research are two-fold. Firstly, we present the tradeoff analysis of performance improvement and energy saving when making offloading decisions. We cut the execution time into three intervals, namely, never offload, tradeoff and always offload. Secondly, we propose and study a novel adaptive offloading model based on the above tradeoff interval. A server on cloud with a critical value of speedup F for a specified mobile device can be found due to the elasticity of cloud computing.

The remainder of this paper is proceeded as follows. Section II presents how to make offloading decisions based on performance improvement and energy saving. Section III analyzes the tradeoff between performance improvement and energy saving. An adaptive offloading scheme is proposed and investigated in Section IV. Section V concludes the paper.

II. Offloading decisions

To begin with, we will provide a brief introduction of mobile cloud computation offloading systems.

A typical architecture of cloud offloading systems is depicted in Figure 1. There are three major components, resource monitoring, cost and partition models, separately.

Resource monitoring model is used to collect resource information such as CPU utilization, battery level, speed and network bandwidth. And the partition model is to cut the composing classes of the application into remote partition and local partition, where the former is offloaded to cloud and the latter is executed locally on the mobile device. The key component is the cost model, where the offloading decision is further made based on a selected cost criterion. There are six criteria listed in Figure 1, on the one side, energy, price cost and storage are cost criteria which are the less the better, and on the other side, performance, robustness and security are benefit criteria which need to be maximize. Among these criteria, energy and performance are the most two important aspects concerned by the mobile users, and thus we will mainly focus on them in the following analysis.

It is obviously that offloading the application from mobile devices onto the remote cloud server can shorten execution time and save energy consumption. However, remote execution is an opportunistic alternative, but not a must, because processing on the cloud requires additional data communication, which may increase the time and the battery consumed by communication. Therefore, offloading decisions should be made when encountering with large communication data or low bandwidth. Is it worth to offload the program from local to the cloud? We will expand it in two ways.

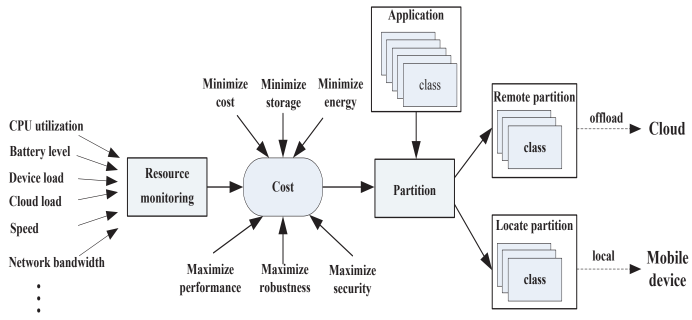


Fig. 1. Architecture of cloud offloading systems

A. Performance improvement

Computation offloading has become an attractive way for reducing execution time required by users, due to applications on mobile devices are becoming increasing intensive, and performance is an important factor to consider. Take the speech recognition of Apple's new product Siri on iPhone 4s [4] for example, the amount of computing is too large to perform on the iPhone 4s and it will take such a long time to get the result that it can't meet the user's need, and thus it should be offloaded to the cloud, in order to save time and improve performance.

We will take execution time saving into consideration when making offloading decisions. The parameters used below are listed in Table 1.

For communication time, we assume that the bandwidth remains the same for the bidirectional communication, and thus it takes $\frac{D}{B}$ seconds to transmit and receive data. The time incurred by offloading is the sum of communication time and computing time on the cloud server and it should be smaller than the execution time on the mobile device in order to improve performance.

TABLE I
PARAMETERS OF OFFLOADING

symbol	meaning
p_m	power for computing
p_i	power while being idle
p_{tr}	power for sending and receiving data
B	network bandwidth
D	exchanged data
t_m	execution time on the mobile device
t_s	execution time on the server

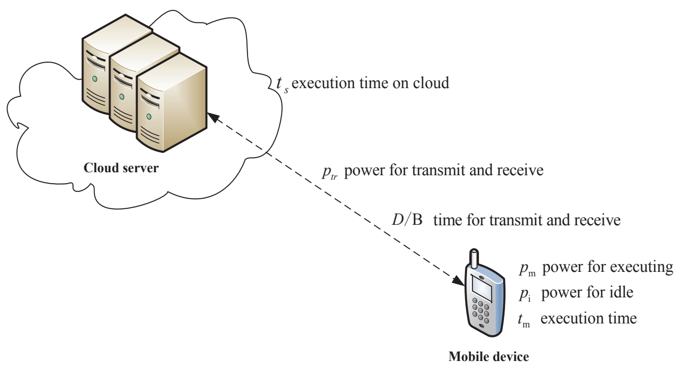
Therefore, it is worth offloading the program to the cloud rather than executing on the local, when it meets the following condition [5]

$$t_m > t_s + \frac{D}{B} \tag{1}$$

From equation (1), it can be seen that offloading can improve performance when execution, including computation and communication, can be performed faster on the cloud than on the mobile device.

B. Energy saving

Energy is another primary aspect that must be considered when making offloading decisions. An investigation engaged by thousands of users around the world indicated that longer battery life to be more important than all other features [6]. Mobile devices such as ipad and iphone are more and more frequently used for watching videos, web surfing, interactive gaming, augmented reality and other purposes which consume huge power and shorten the battery life as a result. Further, these applications are too computation intensive to be executed on a mobile system.



t_s execution time on cloud
Cloud server
p_{tr} power for transmit and receive
D/B time for transmit and receive
p_m power for executing
p_i power for idle
t_m execution time
Mobile device

Fig. 2. Diagram of energy cost during the whole offloading process

According to the diagram of energy cost during the whole offloading process described in Figure 2, it is worth offloading the program to the cloud in order to save energy rather than executing on the local, when [7]

$$p_m t_m > p_i t_s + p_{tr} \frac{D}{B} \tag{2}$$

Similarly, it is found from equation (2) that offloading can save energy when the energy spent on communication and computation on the cloud is much smaller than the energy consumed by the mobile device.

III. Tradeoff analysis

A. Problem definition of tradeoff

In equation (1) and equation (2), for simplicity, let $\frac{D}{B} = C$ and $t_m = F t_s$, where the speedup F indicates how powerful a cloud server is in terms of execution speed comparing with that of the mobile device. Normally, F is much larger than 1 due to the servers are resource-rich while the mobile device is resource-limited.

For allowing offloading to improve performance and save energy at the same time, it has to satisfy the following two conditions

$$t_m > \frac{t_m}{F} + C \tag{3}$$

$$p_m t_m > p_i \frac{t_m}{F} + p_{tr} C \tag{4}$$

The inequalities in (3) and (4) maintain under several conditions: large F that the server is much faster than the mobile device, small D that only a small amount of data is exchanged, and large B that the network bandwidth between the mobile device and the server is high [8].

In order to analyze the relation between remote execution on cloud and local execution on mobile device intuitively, we define the proportion as

$$G = G_1^{\rho} G_2^{1-\rho} \tag{5}$$

where $G_1 = \frac{t_m}{\frac{t_m}{F} + C}$ is the ratio of execution time, $G_2 = \frac{p_m t_m}{p_i \frac{t_m}{F} + p_{tr} C}$ is the ratio of energy cost and ρ is the weight coefficient, and it satisfies $0 \le \rho \le 1$.

The total proportion G is a performance indicator, it considers both energy and time saving at the same time, and the bigger G is, the better the offloading system works.

We first consider two extreme situations here. On the one hand, if the execution time ratio $G_1 = 1$, the proportion $G = G_2^{1-\rho}$ is independent of G_1, where $G_2 = \frac{F p_m}{(F-1) p_{tr} + p_i}$ depends on the power for computing p_m, the power while being idle p_i, the power for sending and receiving data p_{tr} and speedup F. On the other hand, if the energy cost ratio $G_2 = 1$, the proportion $G = G_1^{\rho}$ is not related with G_2, and we have $G_1 = \frac{F p_{tr}}{F p_m + p_{tr} - p_i}$.

We then turn to general circumstances in the middle. For instance, an HP iPAQ PDA with a $400MHz$ Intel XScale processor has the following values: $p_m \approx 0.9W$, $p_i \approx 0.3W$ and $p_{tr} \approx 1.3W$ [7]. We fix execution time on the mobile device $t_m = 2s$ and exchanged data $D = 64KB$. Besides, the weight coefficient ρ is evaluated from 0 to 1 and the speedup F is evaluated from 1 to 21.

Accordingly, the relation between the weight coefficient ρ and the speedup F is illustrated as Figure 3. Here we consider three situations that network bandwidth B is set as $16Kbps$, $64Kbps$ and $128Kbps$, respectively.

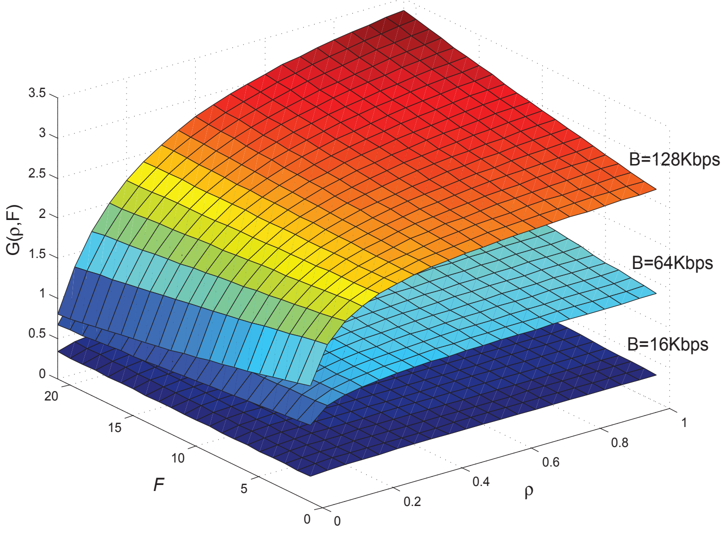


Fig. 3. The relation between the weight coefficient ρ and the speedup F

It can be seen from Figure 3 that the network bandwidth has a huge impact on G, for example, when $B = 16Kbps$, G is always under 1 no matter how F and ρ change, which means offloading the program from mobile device to remote cloud can neither save energy nor reduce time when B is small. But with the increase of B, G is also getting larger, indicating that offloading works better. And also F has the similar effect once ρ is fixed, that is the larger F is, the more energy and time saving can be achieved. Besides, when ρ is small, G rises slowly with the increase of F, however, when ρ is big, G rises faster with the increase of F. Therefore, the weight coefficient ρ should be chosen carefully and it should adjust to the actual offloading systems.

For offloading break-even, that is when t_m arrives at an equilibrium point in inequality (3) or inequality (4). If we assume t_{be1} and t_{be2} to be the critical time values, respectively, we could further derivative the following two equalities

$$t_{be1} = \frac{t_{be1}}{F} + C \Longrightarrow t_{be1} = \frac{C}{1 - \frac{1}{F}} \tag{6}$$

$$p_m t_{be2} = p_i \frac{t_{be2}}{F} + p_{tr} C \Longrightarrow t_{be2} = \frac{p_{tr} C}{p_m - \frac{p_i}{F}} \tag{7}$$

Equation (6) requires that $1 - \frac{1}{F} > 0$, which is $F > 1$, since the time of t_{be1} should be positive. Similarly, equation (7) requires that $p_m - \frac{p_i}{F} > 0$, which can be further transformed as $F > \frac{p_i}{p_m}$, as far as we know that $p_i < p_m$, since the power while being idle must be smaller than the active power of computing. Therefore, these constraints can always be met due to $F > 1$. Especially, when $p_i = p_{tr}$, inequality (4) reduces to

$$t_m > \frac{p_i}{p_m} \left(\frac{t_m}{F} + C \right) \tag{8}$$

From inequality (8), it is found that there is no need to discuss energy saving in inequality (4), as long as it meets performance improvement in inequality (3).

Thus, by using computation offloading to shorten execution time and in the meanwhile to extend battery life of the mobile device, t_m has to meet the following requirement

$$t_m > max(t_{be1}, t_{be2}) \tag{9}$$

Moreover, in order to compare t_{be1} with t_{be2}, let

$$\frac{t_{be1}}{t_{be2}} = \frac{C}{1 - \frac{1}{F}} \cdot \frac{p_m - \frac{p_i}{F}}{p_{tr} C} < 1 \tag{10}$$

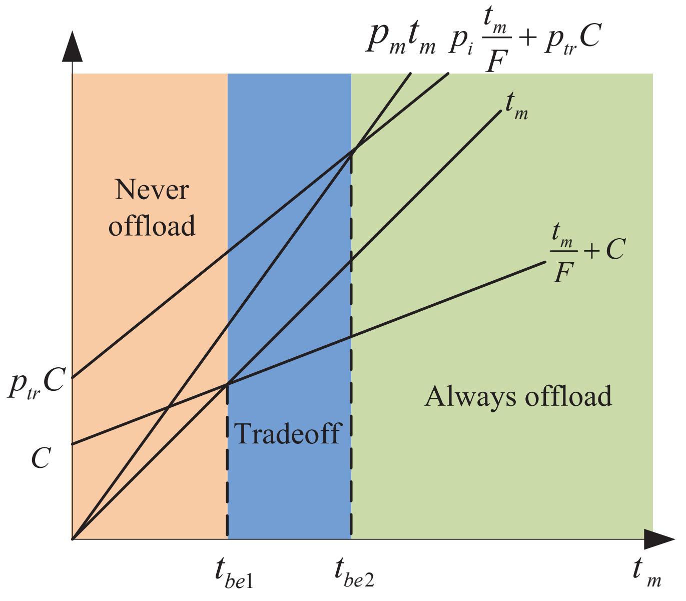
From equation (10), when $F < \frac{p_i - p_{tr}}{p_m - p_{tr}}$, it can be seen that $t_{be1} < t_{be2}$, and otherwise when $F > \frac{p_i - p_{tr}}{p_m - p_{tr}}$ or $p_m = p_{tr}$, we have $t_{be1} > t_{be2}$.

B. Examples of tradeoff

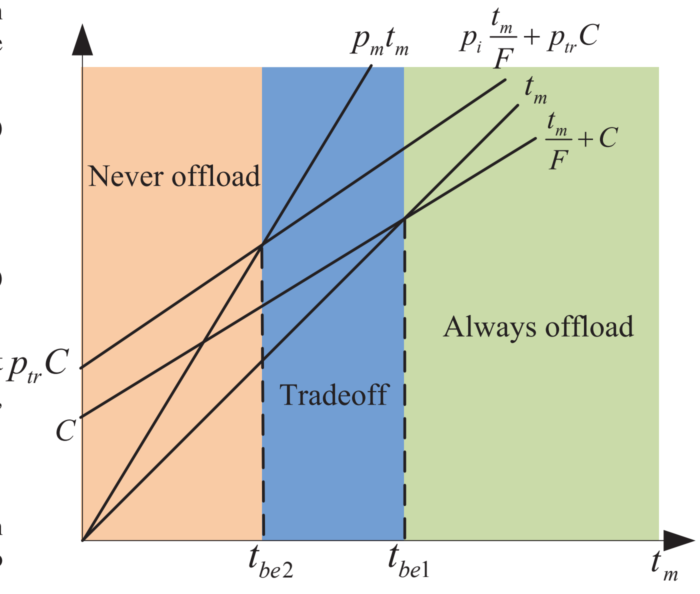
In order to discuss the issue of offloading tradeoff between performance improvement and energy saving in detail, two situations are considered as shown by Figure 4.

As illustrated in Figure 4a (assume $t_{be1} < t_{be2}$), it can be seen that when $t_m < t_{be1}$, offloading is neither beneficial for improving performance nor saving energy, and thus program should never be offloaded to the server in this area. When $t_{be1} < t_m < t_{be2}$, offloading saves time while costs much more energy to execute the program. But when $t_m > t_{be2}$, shifting the complex parts of the program to server on cloud is always beneficial, therefore we should always offload in this interval.

Similarly, as depicted in Figure 4b (assume $t_{be1} > t_{be2}$), it should never offload the program to the server when $t_m < t_{be2}$, but should always offload when $t_m > t_{be1}$. When $t_{be2} < t_m < t_{be1}$, cloud offloading saves energy while costs much more time to execute the program.



(a) $t_{be1} < t_{be2}$



(b) $t_{be1} > t_{be2}$

Fig. 4. Making offloading decisions based on t_m

Therefore, it is the tradeoff between improving performance and saving energy in the interval between t_{be1} and t_{be2}. The values of p_m, p_i and p_{tr} are parameters specific to the mobile system. Besides, the speedup F is determined by the given mobile device and server. The value of F is elastic since different numbers of processors can be obtained from the cloud on demand. Obviously, the large F is, the more resources are needed to ensure such speedup while costing more money. Considering with economic factor, it is not the larger F is, the better the offloading system is. When a certain value of F meets the requirement by the system, it is worth offloading program to such a server on cloud. Furthermore, we can compare F with $\frac{p_i - p_{tr}}{p_m - p_{tr}}$, and thus the relationship between t_{be1} and t_{be2} is known.

IV. Adaptive offloading scheme

Based on the above analysis in Section III, we find that offloading is an optimization method, not a requirement. Therefore, we propose a scheme called adaptive cloud offloading model which is depicted in Figure 5.

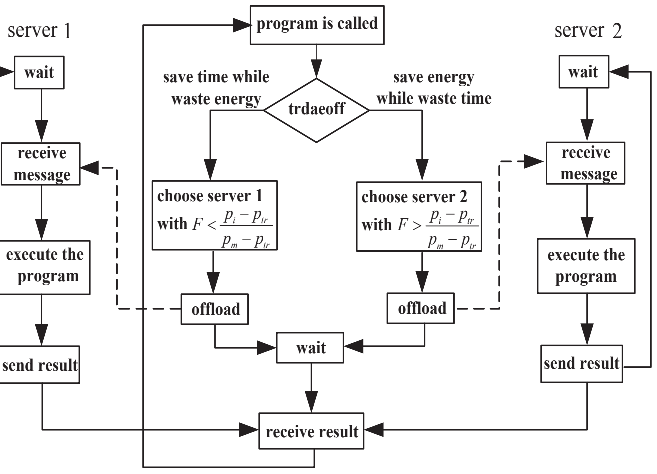

server 1
program is called
server 2
wait
save time while waste energy
save energy while waste time
wait
trdaeoff
receive message
receive message
choose server 1 with $F < \frac{p_i - p_{tr}}{p_m - p_{tr}}$
choose server 2 with $F > \frac{p_i - p_{tr}}{p_m - p_{tr}}$
execute the program
execute the program
offload
offload
wait
send result
send result
receive result

Fig. 5. An adaptive offloading model

From Figure 5, the offloading program is assumed as a function, which could be repeatedly called during the execution of the program. Server 1 and server 2 are on the same cloud, therefore the network distances from the mobile device to the cloud servers stay the same.

When the program is called, a tradeoff decision between time reducing and energy saving should be made before executing the program. For a specified mobile device, $\frac{p_i - p_{tr}}{p_m - p_{tr}}$ is the critical value that the adaption offloading scheme depends on. If the execution time is what we are most concerned about, we should choose the server 1 with speedup $F < \frac{p_i - p_{tr}}{p_m - p_{tr}}$ and the program is offloaded to it, otherwise the server 2 with $F > \frac{p_i - p_{tr}}{p_m - p_{tr}}$ should be chosen. And then, the computation is migrated to the selected server on cloud. Further, the same function is executed on the server and results are sent back to the mobile system [9]. Once received the results, another program is called on the mobile device.

V. Conclusion

To sum up, we argue for tradeoff between time and energy saving with cloud offloading in this paper, while taking economic factor into consideration, which is also the requirement for Green IT. The execution time is divided into three intervals, never offload, tradeoff and always offload based on critical time values. Further, a comprehensive study of tradeoff examples is undertaken according to the three intervals.

Thanks to the elasticity of cloud that the resources can be bought on demand, a proposed adaptive offloading scheme can be realized with the help of cloud computing and it doesn't require estimating the execution time. We have tried to find a server on cloud with a critical value of F for a specified mobile device. Detail experiments will be implemented in our future research.

References

[1] X. Gu, A. Messer, I. Greenberg, D. Milojicic and K. Nahrstedt, *Adaptive offloading for pervasive computing*. IEEE Pervasive Computing Magazine, vol 3, July 2004

[2] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica and M. Zaharia, *Above the clouds: a berkeley view of cloud computing*. Technical Report No. UCB/EECS-2009-28, University of California at Berkley, USA, Feb.10, 2009

[3] A. P. Miettinen and J. K. Nurminen, *Energy efficiency of mobile clients in cloud computing*. HotCloud'10 Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, 2010

[4] http://www.apple.com/iphone/

[5] R. Wolski, S. Gurun, C. Krintz, and D. Nurmi, *Using bandwidth data to make computation offloading decisions*. In Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS 2008), High-Performance Grid Computing Workshop, April 2008

[6] www.cnn.com/2005/TECH/ptech/09/22/phone.study

[7] K. Kumar and Y. Lu, *Cloud computing for mobile users: can offloading computation save energy?* IEEE Computer, vol 43, pp.51-56, April 2010

[8] K. Kumar, J. Liu, Y.-H. Lu and B. Bhargava, *A Survey of Computation Offloading for Mobile Systems*. Mobile Networks and Applications, April 2012

[9] C. Xian, Y.-H. Lu, and Z. Li, *Adaptive computation offloading for energy conservation on battery-powered systems*. In International Conference on Parallel and Distributed Systems, pp.1-8, December 2007